

WEST Search History

[Hide Items](#)[Restore](#)[Clear](#)[Cancel](#)

DATE: Monday, April 05, 2004

Hide?	Set Name	Query	Hit Count
		<i>DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L9	L7 and (insert or inserting)	33
<input type="checkbox"/>	L8	L7 and insert\$	40
<input type="checkbox"/>	L7	L5 and (interrupt\$ or halt\$)	50
<input type="checkbox"/>	L6	L5 and (clean\$ near breakpoint)	0
<input type="checkbox"/>	L5	L3 and (L1 or L2)	57
		<i>DB=EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L4	L3	8
		<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L3	breakpoint near (remov\$ or delet\$ or clean\$)	258
		<i>DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L2	714/32-39.ccls.	2203
<input type="checkbox"/>	L1	717/124-133,154-161.ccls.	2035

END OF SEARCH HISTORY

Hit List

[Clear](#)[Generate Collection](#)[Print](#)[Fwd Refs](#)[Bkwd Refs](#)[Generate OACS](#)

Search Results - Record(s) 1 through 20 of 33 returned.

☐ 1. Document ID: US 20030233634 A1

Using default format because multiple data bases are involved.

L9: Entry 1 of 33

File: PGPB

Dec 18, 2003

PGPUB-DOCUMENT-NUMBER: 20030233634

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030233634 A1

TITLE: Open debugging environment

PUBLICATION-DATE: December 18, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Carrez, Stephane	Issy Les Moulineaux		FR	
Maslov, Guennadi	Rucil Malmaison		FR	
Mirowski, Adam	Le Chesnay		FR	

US-CL-CURRENT: 717/124

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draw. Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	------------	----

☐ 2. Document ID: US 20030196144 A1

L9: Entry 2 of 33

File: PGPB

Oct 16, 2003

PGPUB-DOCUMENT-NUMBER: 20030196144

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030196144 A1

TITLE: Processor condition sensing circuits, systems and methods

PUBLICATION-DATE: October 16, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Swoboda, Gary L.	Sugar Land	TX	US	
Ehlig, Peter N.	Houston	TX	US	

US-CL-CURRENT: 714/34

ABSTRACT:

A data processing device including a semiconductor chip, an electronic processor on-chip and an on-chip condition sensor connected to the electronic processor for analysis of the operations.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-----------	----

☐ 3. Document ID: US 20030188225 A1

L9: Entry 3 of 33

File: PGPB

Oct 2, 2003

PGPUB-DOCUMENT-NUMBER: 20030188225

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030188225 A1

TITLE: Extended "run to" function

PUBLICATION-DATE: October 2, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Bates, Cary Lee	Rochester	MN	US	
Halverson, Steven Gene	Rochester	MN	US	

US-CL-CURRENT: 714/38; 717/129

ABSTRACT:

Methods, apparatus and articles of manufacture for implementing a run-to function for a selected code portion are provided. One embodiment provides a method of debugging code using a debugging program. The method comprises, for a plurality of user-selected statements of the code, automatically setting breakpoints on each of the plurality of user-selected statements; executing the code until a breakpoint set on a first encountered statement of the plurality of user-selected statements is encountered and fired; halting execution at the breakpoint; and displaying a position at which execution halted via a user interface.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-----------	----

☐ 4. Document ID: US 20030149961 A1

L9: Entry 4 of 33

File: PGPB

Aug 7, 2003

PGPUB-DOCUMENT-NUMBER: 20030149961

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030149961 A1

TITLE: Apparatus, method, and program for breakpoint setting

PUBLICATION-DATE: August 7, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Kawai, Masaki	Seto-shi		JP	
Kawamoto, Takuji	Nagoya-shi		JP	

US-CL-CURRENT: 717/129; 717/125

ABSTRACT:

Disclosed is a breakpoint setting apparatus capable of setting a breakpoint without imposing any burden on a programmer. The breakpoint setting apparatus includes an edited-line list manager 115 for managing an address of an edited line in a source code, and a breakpoint setting/disabling sub unit 106 for setting a breakpoint at the address stored in the edited-line list manager 115. The breakpoint setting apparatus automatically sets a breakpoint on each line where the programmer makes an edit without any specific instruction from the programmer.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draws	Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-------	------	----

☐ 5. Document ID: US 20030115576 A1

L9: Entry 5 of 33

File: PGPB

Jun 19, 2003

PGPUB-DOCUMENT-NUMBER: 20030115576

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030115576 A1

TITLE: Breakpoint safety net

PUBLICATION-DATE: June 19, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Bates, Cary Lee	Rochester	MN	US	
Halverson, Steven Gene	Rochester	MN	US	
Santosuosso, John Matthew	Rochester	MN	US	

US-CL-CURRENT: 717/129

ABSTRACT:

Method, apparatus and article of manufacture for debugging code. One embodiment provides a method of debugging code containing a user-specified breakpoint located within a region of the code. The method comprises executing the code, determining whether the execution of the code exits the region of the code without firing the user-specified breakpoint, and if so, halting the execution of the code.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draws	Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-------	------	----

☐ 6. Document ID: US 20030106045 A1

L9: Entry 6 of 33

File: PGPB

Jun 5, 2003

PGPUB-DOCUMENT-NUMBER: 20030106045
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030106045 A1

TITLE: Object-oriented creation breakpoints

PUBLICATION-DATE: June 5, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Arnold, Jeremy Alan	Rochester	MN	US	
Santosuosso, John Matthew	Rochester	MN	US	

US-CL-CURRENT: 717/129

ABSTRACT:

A computer system, program product and method debug an object-oriented computer program by tracking the creation of objects by a plurality of creators (e.g., constructor methods) of a selected class. A user need not separately track each creator. Instead, a debugger identifies each creator and associates breakpoints with all or a user-specified subset of creators to facilitate tracking. Any of the breakpoints may then trigger a halting of execution during debugging. Moreover, in some instances it may be desirable to track the number of creations by all or the subset of the creators for the selected class during program execution until a user-specified condition is satisfied, whereupon program execution is terminated and debugging information is provided to the user.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-----------	----

☐ 7. Document ID: US 20030066053 A1

L9: Entry 7 of 33

File: PGPB

Apr 3, 2003

PGPUB-DOCUMENT-NUMBER: 20030066053
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030066053 A1

TITLE: SQL debugging using stored procedures

PUBLICATION-DATE: April 3, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Al-Azzawe, Abdul Hamid	San Jose	CA	US	

US-CL-CURRENT: 717/127; 707/100, 709/224

ABSTRACT:

A method, apparatus and article of manufacture is provided for SQL debugging within a computer system network. The method uses stored procedures via a console for debugging of SQL instructions located in a server, wherein preferably only one database communication line exists between the server and the console. The server has a database management system for retrieving data from a database stored in an electronic storage device coupled to the server. The method uses a debugger manager at the console for commanding and monitoring debugging operations of the server-side SQL instructions performed by a debugger engine, and uses stored procedures of a debugger router as a database communication interface for receiving commands and sending status reports between the debugger manager and the debugger engine.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-----------	----

☐ 8. Document ID: US 20030046668 A1

L9: Entry 8 of 33

File: PGPB

Mar 6, 2003

PGPUB-DOCUMENT-NUMBER: 20030046668

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030046668 A1

TITLE: System, method and article of manufacture for distributing IP cores

PUBLICATION-DATE: March 6, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Bowen, Matt	Littlemore		DE	

US-CL-CURRENT: 717/131

ABSTRACT:

A system, method and article of manufacture are provided for distributing cores. In general, a core that includes a plurality of first variables is identified without reference to one or more parameters. A computer program is executed that includes a plurality of second variables with reference to the one or more parameters. The execution of the computer program includes execution of the core. The one or more parameters of the first variables are then inferred from the one or more parameters of the second variables.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-----------	----

☐ 9. Document ID: US 20030005415 A1

L9: Entry 9 of 33

File: PGPB

Jan 2, 2003

PGPUB-DOCUMENT-NUMBER: 20030005415

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20030005415 A1

TITLE: Low impact breakpoint for multi-user debugging

PUBLICATION-DATE: January 2, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Bates, Cary Lee	Rochester	MN	US	
Schmidt, William Jon	Rochester	MN	US	

US-CL-CURRENT: 717/129

ABSTRACT:

Method and system for a software debugger tool. Breakpoints are submitted, as breakpoint data, by a user. A breakpoint manager stores the breakpoint data and inserts the breakpoints into the software program code. The breakpoint manager gains control of the program when a breakpoint is processed associated with a particular job. After the breakpoint manager completes an interrupt routine to process the breakpoint, using instructions stored in the breakpoint data, the method removes breakpoints associated with the particular job. When control is to be returned to the program, only those breakpoints that are found to be useful are set.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	RWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-----------	----

☐ 10. Document ID: US 20020199173 A1

L9: Entry 10 of 33

File: PGPB

Dec 26, 2002

PGPUB-DOCUMENT-NUMBER: 20020199173

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20020199173 A1

TITLE: System, method and article of manufacture for a debugger capable of operating across multiple threads and lock domains

PUBLICATION-DATE: December 26, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Bowen, Matt	Oxford		GB	

US-CL-CURRENT: 717/129; 717/131

ABSTRACT:

A system, method and article of manufacture are provided for debugging a computer program. In general, a plurality of threads is identified in a computer program. Selection of one of the threads is allowed. The selected thread is then debugged.

☐ 11. Document ID: US 20020073402 A1

L9: Entry 11 of 33

File: PGPB

Jun 13, 2002

PGPUB-DOCUMENT-NUMBER: 20020073402

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20020073402 A1

TITLE: Method for inserting global breakpoints

PUBLICATION-DATE: June 13, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Sangavarapu, Vamsi Krishna	Bangalore		IN	
Bhattacharya, Suparna	Bangalore		IN	
Krishnamoorthy, Anand	Bangalore		IN	

US-CL-CURRENT: 717/129; 714/45, 717/130

ABSTRACT:

A method, an apparatus, and a computer program product for inserting one or more global breakpoints for debugging computer software are disclosed. A method, an apparatus, and a computer program product for removing one or more global breakpoints for debugging computer software are also disclosed. The inserting method includes the steps of: inserting a global breakpoint in a page containing software code if the page is present in memory; reading the page into memory if not present in memory, and inserting a global breakpoint in the page immediately after being read into memory; and detecting a private copy of the page if present, and inserting a global breakpoint in the private copy.

☐ 12. Document ID: US 20020073401 A1

L9: Entry 12 of 33

File: PGPB

Jun 13, 2002

PGPUB-DOCUMENT-NUMBER: 20020073401

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20020073401 A1

TITLE: Method of detecting zombie breakpoints

PUBLICATION-DATE: June 13, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Sangavarapu, Vamsi Krishna	Bangalore		IN	

Bhattacharya, Suparna
Moore, Richard J.

Bangalore
Waterlooville

IN
GB

US-CL-CURRENT: 717/129; 714/45, 717/126

ABSTRACT:

A method, an apparatus, and a computer program product for detecting one or more zombie global breakpoints for debugging computer software are disclosed. The method includes the steps of: checking a breakpoint data structure to determine if a breakpoint known to a debugging process is at an address where a breakpoint fired; if a known breakpoint cannot be determined at the address, verifying if a breakpoint condition continues to exist at the address where the breakpoint fired; and if the breakpoint condition does not exist, identifying the breakpoint as a zombie breakpoint.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Sequen	Attachments	Claims	KWIC	Draw Desc	Fig
------	-------	----------	-------	--------	----------------	------	-----------	--------	-------------	--------	------	-----------	-----

☐ 13. Document ID: US 6587969 B1

L9: Entry 13 of 33

File: USPT

Jul 1, 2003

US-PAT-NO: 6587969

DOCUMENT-IDENTIFIER: US 6587969 B1

TITLE: Software system and methods for testing the functionality of a transactional server

DATE-ISSUED: July 1, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Weinberg; Amir	Tsoran			IL
Leshem; Eran	Gan Shomron			IL
Kholmyansky; Maxim	Tel-Aviv			IL
Garri; Amos	Modiin			IL
Tapiro; Nisim	Letsion			IL
Hillel; Meni	San Jose	CA		

US-CL-CURRENT: 714/46; 714/38

ABSTRACT:

A testing tool automatically records a series of user steps taken during a user session with a transactional server and generates a test for testing the functionality of server. Through a user interface of the testing tool, the user can define verification steps to automatically test for expected server responses during test execution. The testing tool displays the test to the user as a tree having nodes (displayed as icons) which represent steps of the test. Via the user interface, the user can modify node properties and perform other types of tree edit operations to edit the test, without the need to know a scripting or other programming language. When the user selects a node that corresponds to a particular field or other object of the server screen, the testing tool automatically displays the screen with the object highlighted. The testing tool

also allows the test author to use a spreadsheet to conveniently specify data sets for running multiple iterations of a test; thus, the user can record a single transaction and then automatically test the transaction with other data sets.

55 Claims, 23 Drawing figures
Exemplary Claim Number: 40
Number of Drawing Sheets: 22

Full	Title	Citation	Front	Review	Classification	Date	Reference			Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--	--	--------	------	-----------	----

☐ 14. Document ID: US 6493868 B1

L9: Entry 14 of 33

File: USPT

Dec 10, 2002

US-PAT-NO: 6493868
DOCUMENT-IDENTIFIER: US 6493868 B1

TITLE: Integrated development tool

DATE-ISSUED: December 10, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
DaSilva; Greg N.	Rexdale			CA
Gingrich; Paul	Toronto			CA
Pandey; Raju	Toronto			CA

US-CL-CURRENT: 717/105; 345/1.3, 345/540, 717/108, 717/111, 717/129, 717/130

ABSTRACT:

An integrated code development tool, comprising of an editor, a project management and build system, a debugger, a profiler, and a graphical data visualization system. The editor is operable to provide a source code view which is simultaneously capable of integrating with said debugger to provide for stepping through code and setting breakpoints, and integrating with the output of said build system to display source code interleaved with corresponding assembler code created by said build system.

19 Claims, 8 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 4

Full	Title	Citation	Front	Review	Classification	Date	Reference			Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--	--	--------	------	-----------	----

☐ 15. Document ID: US 6412106 B1

L9: Entry 15 of 33

File: USPT

Jun 25, 2002

US-PAT-NO: 6412106
DOCUMENT-IDENTIFIER: US 6412106 B1

TITLE: Graphical system and method for debugging computer programs

DATE-ISSUED: June 25, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Leask; Gary M.	Dallas	TX		
Huffman; Dale L.	Allen	TX		

US-CL-CURRENT: 717/124; 717/125, 717/148

ABSTRACT:

A system and method for graphically debugging a computer program is disclosed. In a preferred embodiment, a graphical debugging environment is provided, which is capable of displaying a graphical representation of an application program to be debugged. Thereafter, the graphical debugging environment allows a user to insert debugging tools, such as breakpoints, directly into the graphical representation of the application program. Thus, a user is not required to interact with the textual source code of an application program when debugging it. The graphical debugging environment may display indicators illustrating where debug tools have been inserted within the application program. In a preferred embodiment, the graphical debugging environment allows a user to perform debugging during an application program's runtime. Thus, a user is not required to halt an application program prior to debugging it. Also, in a preferred embodiment the graphical debugging environment executing on a local computer may be used to debug an application program residing on a remote computer.

52 Claims, 9 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 7

Full	Title	Citation	Front	Review	Classification	Date	Reference			Claims	KWIC	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--	--	--------	------	-----------	----

☐ 16. Document ID: US 6330691 B1

L9: Entry 16 of 33

File: USPT

Dec 11, 2001

US-PAT-NO: 6330691

DOCUMENT-IDENTIFIER: US 6330691 B1

TITLE: Use of dynamic translation to provide breakpoints in non-writeable object code

DATE-ISSUED: December 11, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Buzbee; William B.	Half Moon Bay	CA		
Burch; Carl D.	Mountain View	CA		

US-CL-CURRENT: 714/35; 714/38

ABSTRACT:

Dynamic translation is used during debugging of a computer application process. The computer application process resides in a computing system in which blocks of code within a shared library are utilized by the computer application process. The blocks of code within the shared library are also available to be utilized by other applications in the system. During runtime, the computer application process is dynamically translated to produce translated code. The dynamic translation includes translation of a first block of code within the shared library to produce a translated block of code. The translated block of code is included within the translated code. Debugging code, such as a break instruction, may then be added to the translated code. Alternatively, only blocks of code within the shared library which are called by the computer application process or modified by a debugger are dynamically translated. In this alternative case, the object code for the computer application process is executed without being dynamically translated.

14 Claims, 12 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 12

Full	Title	Citation	Front	Review	Classification	Date	Reference	Abstract	Claims	Drawings	Draw. Desc.	Inventor
------	-------	----------	-------	--------	----------------	------	-----------	----------	--------	----------	-------------	----------

☐ 17. Document ID: US 6243857 B1

L9: Entry 17 of 33

File: USPT

Jun 5, 2001

US-PAT-NO: 6243857
DOCUMENT-IDENTIFIER: US 6243857 B1

TITLE: Windows-based flowcharting and code generation system

DATE-ISSUED: June 5, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Logan, III; Frank G.	Ann Arbor	MI		
Bunch; Kenneth W.	Newport News	VA		
Davis; Teddy Martin	Norfolk	VA		
Achesinski; Jeffrey M.	Virginia Beach	VA		

US-CL-CURRENT: 717/111; 717/113, 717/125

ABSTRACT:

A machine control system (130) includes a computer (132) that generates, edits and displays a continuous multi-block flowchart representing a program and compiles the program from the flowchart to control the operations of a machine (140). The system (130) also includes a debugger (146) for displaying the flowchart in a debugger window (170) for runtime execution control of the program.

18 Claims, 11 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 9

☐ 18. Document ID: US 6158045 A

L9: Entry 18 of 33

File: USPT

Dec 5, 2000

US-PAT-NO: 6158045

DOCUMENT-IDENTIFIER: US 6158045 A

TITLE: Portable debugging services utilizing a client debugger object and a server debugger object with flexible addressing support

DATE-ISSUED: December 5, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
You; Lawrence L.	San Jose	CA		

US-CL-CURRENT: 717/124

ABSTRACT:

A set of portable services for debugging computer software programs is disclosed. The services provide an object-oriented programming framework which is portable to various hardware and operating system platforms. The framework consists primarily of a debugger server and a debugger client. Multiple debugger clients, which target multiple processes executing on heterogeneous systems, can be used concurrently from a single high-level debugger process. Clients can processes locally and remotely. An addressing abstraction is utilized to facilitate the use of target memory addresses in a portable fashion. The use of static compiler types such as void* in the C and C++ languages do not properly express this as a portable abstraction since they are limited to the static size as implemented by the compiler used to compile the debugger code. Two major classes are used to describe the abstraction itself; other subclasses are created to describe processor-specific addressing data.

12 Claims, 18 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 13

☐ 19. Document ID: US 6077312 A

L9: Entry 19 of 33

File: USPT

Jun 20, 2000

US-PAT-NO: 6077312

DOCUMENT-IDENTIFIER: US 6077312 A

TITLE: Apparatus, program product and method of debugging utilizing a context sensitive breakpoint

DATE-ISSUED: June 20, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bates; Cary Lee	Rochester	MN		
Day; Paul Reuben	Rochester	MN		

US-CL-CURRENT: 717/129; 717/131, 717/134

ABSTRACT:

An apparatus, program product, and method of debugging a computer program utilize a context sensitive breakpoint to conditionally halt execution of a computer program when the context of the computer program meets a predetermined criteria. The context of a computer program is defined by a calling history associated with the computer program that identifies, at any given instant during the execution of the computer program, the sequence of routines in the computer program that were called prior to reaching the instruction being processed at that given instant. By basing the determination of whether to interrupt processing of a computer program upon the calling history for the computer program, a computer programmer is given a great deal of flexibility in setting breakpoints that isolate the circumstances for which it is desired to induce stoppage of a computer program from other unwanted circumstances.

27 Claims, 13 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 6

Full	Title	Citation	Front	Review	Classification	Date	Reference		Claims	RMK	Draw Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--	--------	-----	-----------	----

☐ 20. Document ID: US 6032268 A

L9: Entry 20 of 33

File: USPT

Feb 29, 2000

US-PAT-NO: 6032268

DOCUMENT-IDENTIFIER: US 6032268 A

TITLE: Processor condition sensing circuits, systems and methods

DATE-ISSUED: February 29, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Swoboda; Gary L.	Sugar Land	TX		
Ehlig; Peter N.	Houston	TX		

US-CL-CURRENT: 714/30; 714/34, 714/726

ABSTRACT:

The invention provides improved architectures and methods for emulation, simulation, and testability of data processing devices and systems without requiring physical probing or special test fixtures. A data processing device may include a semiconductor chip that is

divided into domains. One domain may be halted and tested while another domain continues to operate. For example, the semiconductor chip may have a electronic processor domain and an analysis domain. The analysis domain may include an on-chip condition sensor that is connected to the electronic processor domain. The chip can further include control logic circuitry to allow the analysis domain to operate while the electronic processor is halted and tested.

18 Claims, 47 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 41

Full	Title	Citation	Front	Review	Classification	Date	Reference			Claims	KWIC	Draw. Desc	In
------	-------	----------	-------	--------	----------------	------	-----------	--	--	--------	------	------------	----

Clear

Generate Collection

Print

Fwd Refs

Bkwd Refs

Generate OACS

Terms

Documents

L7 and (insert or inserting)

33

Display Format:

Change Format

[Previous Page](#)

[Next Page](#)

[Go to Doc#](#)

Hit List

[Clear](#)[Generate Collection](#)[Print](#)[Fwd Refs](#)[Bkwd Refs](#)[Generate OACS](#)

Search Results - Record(s) 1 through 8 of 8 returned.

☐ 1. Document ID: NNRD454139

Using default format because multiple data bases are involved.

L4: Entry 1 of 8

File: TDBD

Feb 1, 2002

TDB-ACC-NO: NNRD454139

DISCLOSURE TITLE: Code Coverage Tool with Call Site Procedural Statistics

PUBLICATION-DATA:

IBM technical Disclosure Bulletin, February 2002, UK

ISSUE NUMBER: 454

PAGE NUMBER: 306

SECURITY: Use, copying and distribution of this data is subject to the restrictions in the Agreement For IBM TDB Database and Related Computer Databases. Unpublished - all rights reserved under the Copyright Laws of the United States. Contains confidential commercial information of IBM exempt from FOIA disclosure per 5 U.S.C. 552(b)(4) and protected under the Trade Secrets Act, 18 U.S.C. 1905.

COPYRIGHT STATEMENT: The text of this article is Copyrighted (c) IBM Corporation 2002. All rights reserved.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Summary	Abstract	Claims	Keywords	Drawings
------	-------	----------	-------	--------	----------------	------	-----------	---------	----------	--------	----------	----------

☐ 2. Document ID: NNRD454132

L4: Entry 2 of 8

File: TDBD

Feb 1, 2002

TDB-ACC-NO: NNRD454132

DISCLOSURE TITLE: Code Coverage Tool with Reentrant Indication

PUBLICATION-DATA:

IBM technical Disclosure Bulletin, February 2002, UK

ISSUE NUMBER: 454

PAGE NUMBER: 299

DISCLOSURE TEXT:

This invention provides an indication within a code coverage report as to whether or not a particular routine has been called in a recursive manner. Today, one way that many code coverage tools (such as the ATC code coverage tool) work is by using the debugger support to set a breakpoint on each statement. As the statements are encountered the fact that they were hit is recorded. After this, the breakpoint can be removed. This invention uses a similar concept to detect if a routine is called recursively. The

invention handles the breakpoint set on an entry point to a routine differently than the rest of the breakpoints set to gather coverage data. When the breakpoint on the entry point is hit, the invention will examine the call stack and conclude if another instance of this routine is currently active on the stack. If so, it records the fact that this routine was recursively invoked. Then the breakpoint is removed. If the routine was not recursively called, then the fact the statement was executed is recorded but the breakpoint is not removed since a recursive version of the call has not yet been detected. Recursion depth is another statistic that can be reported by the tool. To do this the breakpoint that tests for recursion is not removed so that one can get a cumulative picture of the recursion taking place in the program. The when the breakpoint that tracks recursion fires the it check the recursion depth and includes the maximum recursion depth in the report. In the example ATC report below the function foo() was called recursively.

SECURITY: Use, copying and distribution of this data is subject to the restrictions in the Agreement For IBM TDB Database and Related Computer Databases. Unpublished - all rights reserved under the Copyright Laws of the United States. Contains confidential commercial information of IBM exempt from FOIA disclosure per 5 U.S.C. 552(b)(4) and protected under the Trade Secrets Act, 18 U.S.C. 1905.

COPYRIGHT STATEMENT: The text of this article is Copyrighted (c) IBM Corporation 2002. All rights reserved.

Full	Title	Citation	Front	Review	Classification	Date	Reference			Claims	KWIC	Draw Desc
------	-------	----------	-------	--------	----------------	------	-----------	--	--	--------	------	-----------

☐ 3. Document ID: NN8903238

L4: Entry 3 of 8

File: TDBD

Mar 1, 1989

TDB-ACC-NO: NN8903238

DISCLOSURE TITLE: Icon-Editing Semantics in XDE

PUBLICATION-DATA:

IBM Technical Disclosure Bulletin, March 1989, US

VOLUME NUMBER: 31

ISSUE NUMBER: 10

PAGE NUMBER: 238 - 239

DISCLOSURE TEXT:

- Icon editing is a useful and unique feature of the XDE software development environment. It is convenient for the user to be able to edit the icons by using the same methods applied to editing text. XDE is an advanced software development environment targeted for the IBM RT PC* running IBM's proprietary UNIX** operating system AIX***. This environment presents a variety of windows, each displaying textual information. One of these windows, the source window, displays the application source text corresponding to the currently executing object, or other source that is requested by the user. This window can also be used to set breakpoints or perform various debugging operations. XDE utilizes various icons to convey certain information to the user. For example, stop signs next to the source line indicate that a breakpoint is set at that line. A yield sign indicates that a conditional breakpoint is set at that line. An arrow indicates that the line is the next to be executed. These icons may be edited with the same operations that are used to edit text with appropriate semantics applied to each edit. For example, these icons may be selected in the same manner as text by holding the mouse button down and dragging the mouse cursor over the icon. The icon will be highlighted just as selected text is highlighted. The icon may be cut or copied to an internal buffer and later pasted to a new location. The semantics of cutting or deleting

the stop or yield sign icons is that the breakpoint is removed from the application and the debugging session. When pasted or copied to another location, the semantics dictate that all of the attributes and command sequences associated with the breakpoint accompany the breakpoint at the new location. The semantics associated with deleting the arrow is that execution will resume from the previous call sight of the currently active procedure. Cutting or pasting the icon onto a new location has the effect of changing the location of the next statement to be executed. If the arrow is pasted outside of the currently active procedure, the results will be unpredictable. Icon editing with the appropriate semantics, as described above, is unique to interactive software development environments by allowing the user to manipulate and edit icons in the same manner as text. * Trademark of IBM Corp. ** Trademark of AT&T Bell Laboratories. *** Trademark of IBM Corp.

SECURITY: Use, copying and distribution of this data is subject to the restrictions in the Agreement For IBM TDB Database and Related Computer Databases. Unpublished - all rights reserved under the Copyright Laws of the United States. Contains confidential commercial information of IBM exempt from FOIA disclosure per 5 U.S.C. 552(b)(4) and protected under the Trade Secrets Act, 18 U.S.C. 1905.

COPYRIGHT STATEMENT: The text of this article is Copyrighted (c) IBM Corporation 1989. All rights reserved.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Abstract	Indexing	Claims	RMIC	Draw. Desc
------	-------	----------	-------	--------	----------------	------	-----------	----------	----------	--------	------	------------

☐ 4. Document ID: NN81071192

L4: Entry 4 of 8

File: TDBD

Jul 1, 1981

TDB-ACC-NO: NN81071192

DISCLOSURE TITLE: Permanent Breakpoints Using Hardware Trace. July 1981.

PUBLICATION-DATA:

IBM Technical Disclosure Bulletin, July 1981, US

VOLUME NUMBER: 24

ISSUE NUMBER: 2

PAGE NUMBER: 1192

DISCLOSURE TEXT:

1p. This mechanism provides permanent program breakpoints in a relatively simple manner and without performance degradation. - An accepted method of providing software breakpoints is to replace the first two bytes of an instruction with a supervisor call (SVC) instruction while saving the replaced two bytes. When the SVC instruction is encountered during program execution, the debugging facility will gain control. After breakpoint, processing is completed, the SVC instruction is replaced by the original two bytes and program execution resumes. Unfortunately, this means that the breakpoint is removed after one use. - The new mechanism described herein provides permanent breakpoints. After gaining control in the SVC routine and before allowing execution of the problem program to resume, this new mechanism does the following: It sets a first time switch in the debug program to zero and turns on the hardware trace bit in the level status register contained in the saved level status block of the problem program. When program execution resumes, a hardware trace interrupt will occur. If the first time switch is equal to zero, this means the restored instruction is not about to be executed. In this case, the location of the instruction is saved, the first time switch is set to 1 and execution is resumed with the hardware trace still on for the problem program, but with the inhibit trace bit on in the level status block. This will cause the subject instruction to be executed, and a trace interrupt to occur before the next

instruction. - If at occurrence of the hardware trace interrupt the first time switch is equal to 1, this means that the restored instruction has just been executed. In this case, using the saved location counter, the SVC instruction is put back into the problem program, thus reestablishing the breakpoint. Also, the first time switch is set to zero, and the hardware trace bit in the level status register contained in the saved level status block of the problem program is turned off. Normal execution is then resumed. - Thus, to achieve permanent breakpoints, two trace interrupts are taken following the initial SVC that caused the debugging program to get control at the breakpoint. After the second trace interrupt, the break-point is restored and the trace bit is turned off. This is an easily implemented, low overhead method of achieving permanent breakpoints in a program without either running interpretively or tracing every instruction.

SECURITY: Use, copying and distribution of this data is subject to the restrictions in the Agreement For IBM TDB Database and Related Computer Databases. Unpublished - all rights reserved under the Copyright Laws of the United States. Contains confidential commercial information of IBM exempt from FOIA disclosure per 5 U.S.C. 552(b)(4) and protected under the Trade Secrets Act, 18 U.S.C. 1905.

COPYRIGHT STATEMENT: The text of this article is Copyrighted (c) IBM Corporation 1981. All rights reserved.

Full	Title	Citation	Front	Review	Classification	Date	Reference			Claims	KNOW	Draw Desc
------	-------	----------	-------	--------	----------------	------	-----------	--	--	--------	------	-----------

☐ 5. Document ID: US 20030028862 A1

L4: Entry 5 of 8

File: DWPI

Feb 6, 2003

DERWENT-ACC-NO: 2003-392282

DERWENT-WEEK: 200337

COPYRIGHT 2004 DERWENT INFORMATION LTD

TITLE: Debugger program removes breakpoint based on induction rate, when final value of breakpoint is satisfied and induction variable has a preset value that exceeds final value upon next iteration of loop

INVENTOR: BATES, C L; SCHMIDT, W J

PRIORITY-DATA: 2001US-0918573 (August 1, 2001)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 20030028862 A1	February 6, 2003		008	G06F009/44

INT-CL (IPC): G06 F 9/44

ABSTRACTED-PUB-NO: US20030028862A

BASIC-ABSTRACT:

NOVELTY - An induction rate is acquired from a program code within a loop. A breakpoint is removed based on the acquired induction rate, when the acquired final value of the breakpoint is satisfied and the induction variable has a preset value that exceeds final value upon a next iteration of a loop.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

(1) debugger impact reducing method; and

(2) recorded medium storing program for reducing debugger impact.

USE - Debugger program for reducing debugger impact through motion of breakpoint set.

ADVANTAGE - Reduces debugger impact through motion of break point set effectively.
Reduces program timing errors.

DESCRIPTION OF DRAWING(S) - The figure shows the flowchart explaining debugger impact reduction process.

Full	Title	Citation	Front	Review	Classification	Date	Reference			Claims	RMIC	Draw Desc	C
------	-------	----------	-------	--------	----------------	------	-----------	--	--	--------	------	-----------	---

☐ 6. Document ID: US 20030005415 A1

L4: Entry 6 of 8

File: DWPI

Jan 2, 2003

DERWENT-ACC-NO: 2003-353007

DERWENT-WEEK: 200333

COPYRIGHT 2004 DERWENT INFORMATION LTD

TITLE: Software program debugging method in computer system, involves removing breakpoints associated with particular job and reestablishing removed breakpoints, when breakpoints are determined to be useful

INVENTOR: BATES, C L; SCHMIDT, W J

PRIORITY-DATA: 2001US-0893385 (June 27, 2001)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
<u>US 20030005415 A1</u>	January 2, 2003		023	G06F009/44

INT-CL (IPC): G06 F 9/44

ABSTRACTED-PUB-NO: US20030005415A

BASIC-ABSTRACT:

NOVELTY - An useful breakpoint existing in a program for a particular job, is determined when a control of the program is received by a debugger. The breakpoints except the useful breakpoint, associated with the particular job is removed from the program. The removed breakpoints are reestablished when the breakpoints are determined to be useful before returning control of the program from the debugger.

DETAILED DESCRIPTION - An INDEPENDENT CLAIM is included for computer readable medium storing software program.

USE - For debugging software program in computer system such as hand-held devices, multi-processor system, microprocessor-based or programmable consumer electronics, minicomputers, mainframe computers, etc.

ADVANTAGE - The breakpoints are efficiently processed without degrading the performance of the computer system and the debugging of the software program is easily realized.

DESCRIPTION OF DRAWING(S) - The figure shows the software environment for the computer system.

☐ 7. Document ID: US 20030009745 A1

L4: Entry 7 of 8

File: DWPI

Jan 9, 2003

DERWENT-ACC-NO: 2003-267158

DERWENT-WEEK: 200326

COPYRIGHT 2004 DERWENT INFORMATION LTD

TITLE: Program debugger for computer system, removes initial conditional breakpoint within program loop, if reset breakpoint which includes extracted Boolean expression is satisfied

INVENTOR: BATES, C L; SCHMIDT, W J

PRIORITY-DATA: 2001US-0897608 (July 3, 2001)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 20030009745 A1	January 9, 2003		014	G06F009/44

INT-CL (IPC): G06 F 9/44

ABSTRACTED-PUB-NO: US20030009745A

BASIC-ABSTRACT:

NOVELTY - A Boolean expression is extracted from an initial conditional breakpoint (ICB) within a program loop. A reset breakpoint that includes the extracted Boolean expression, is set at a pre-ICB program position. The initial conditional breakpoint is removed if the reset breakpoint is satisfied.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are included for the following:

- (1) method for reducing debugger impact; and
- (2) computer readable medium storing debugger impact reduction program.

USE - For computer system.

ADVANTAGE - The setting of reset breakpoint at pre-ICB program location, avoids unnecessary evaluation conditional breakpoint within program loop.

DESCRIPTION OF DRAWING(S) - The figure shows the flowchart illustrating the debugger impact reduction program.

☐ 8. Document ID: US 20020073402 A1

L4: Entry 8 of 8

File: DWPI

Jun 13, 2002

DERWENT-ACC-NO: 2002-557150

DERWENT-WEEK: 200259

TITLE: Global breakpoint insertion method for debugging software, involves detecting private copy of page after reading from memory and inserting global breakpoint in detected private copy

INVENTOR: BHATTACHARYA, S; KRISHNAMOORTHY, A ; SANGAVARAPU, V K

PRIORITY-DATA: 2000US-0732342 (December 7, 2000)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 20020073402 A1	June 13, 2002		013	G06F009/44

INT-CL (IPC): G06 F 9/44

ABSTRACTED-PUB-NO: US20020073402A

BASIC-ABSTRACT:

NOVELTY - A global breakpoint is inserted in a page containing software code after reading the page in a memory. The global breakpoint is inserted in the detected private copy of the page.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are included for the following:

- (1) Computer-implemented apparatus for inserting global breakpoint;
- (2) Computer program product for inserting global breakpoints;
- (3) Global breakpoint removal method;
- (4) Computer-implemented apparatus for removing global breakpoint; and
- (5) Computer program product of removing global breakpoint.

USE - For debugging computer software in operating system such as Linux e.g. SUN, HP and AIX for computer system, connected to LAN, WAN, Internet, intranet.

ADVANTAGE - Enables handling elegantly, consistently and seamlessly of the problem of inserting and removing global breakpoints with minimum overhead, hence facilitates the OS to maintain sufficient information effectively.

DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of the operating system.

Full	Title	Citation	Front	Review	Classification	Date	Reference	Abstract	Claims	RMIC	Draw Desc	C
------	-------	----------	-------	--------	----------------	------	-----------	----------	--------	------	-----------	---

Clear	Generate Collection	Print	Fwd Refs	Bkwd Refs	Generate OACS
-------	---------------------	-------	----------	-----------	---------------

Terms	Documents
L3	8

WEST Search History

Hide Items

Restore

Clear

Cancel

DATE: Monday, April 05, 2004

Hide?	Set Name	Query	Hit Count
		<i>DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L11	breakpoint removal	4
<input type="checkbox"/>	L10	breakpoint remover	0
<input type="checkbox"/>	L9	L7 and (insert or inserting)	33
<input type="checkbox"/>	L8	L7 and insert\$	40
<input type="checkbox"/>	L7	L5 and (interrupt\$ or halt\$)	50
<input type="checkbox"/>	L6	L5 and (clean\$ near breakpoint)	0
<input type="checkbox"/>	L5	L3 and (L1 or L2)	57
		<i>DB=EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L4	L3	8
		<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L3	breakpoint near (remov\$ or delet\$ or clean\$)	258
		<i>DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ</i>	
<input type="checkbox"/>	L2	714/32-39.ccls.	2203
<input type="checkbox"/>	L1	717/124-133,154-161.ccls.	2035

END OF SEARCH HISTORY



Welcome
United States Patent and Trademark Office



» Search Re

[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)
[Quick Links](#)

Welcome to IEEE Xplore®

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

Your search matched **1** of **1015452** documents.

A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance** in **Descending** order.

Refine This Search:

You may refine your search by editing the current search expression or entering a new one in the text box.

☐ Check to search within this result set

Results Key:

JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

1 Variable frame rate parameter encoding via adaptive frame selection using dynamic programming

George, E.B.; McCree, A.V.; Viswanathan, V.R.;

Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference

Proceedings., 1996 IEEE International Conference on , Volume: 1 , 7-10 May 1996
Pages:271 - 274 vol. 1

[\[Abstract\]](#)
[\[PDF Full-Text \(404 KB\)\]](#)
[IEEE CNF](#)

Searching for **PHRASE** breakpoint remove.

Restrict to: [Header](#) [Title](#) Order by: [Expected citations](#) [Hubs](#) [Usage](#) [Date](#) Try: [Amazon](#) [B&N](#) [Google \(RI\)](#) [Google \(Web\)](#) [CSB](#) [DBLP](#)

No documents match Boolean query. Trying non-Boolean relevance query.

1000 documents found. Only retrieving 250 documents (System busy - maximum reduced). Retrieving documents... Order: relevance to query.

[Causal Distributed Breakpoints - Fowler, Zwaenepoel \(1990\) \(Correct\) \(39 citations\)](#)

Causal Distributed **Breakpoints** Jerry Fowler Willy Zwaenepoel Department of Texas 772511892 Abstract A causal distributed **breakpoint** is initiated by a sequential **breakpoint** in www.cs.rice.edu/~willy/papers/icdcs90a.ps.gz

[Breakpoint Detection Using Covariance Propagation - Ji, Haralick \(Correct\)](#)

Breakpoint Detection Using Covariance Propagation Qiang Ji presents a statistical approach for detecting **breakpoints** from chain encoded digital arcs. An arc point rate increases. Increasing the included angles **removes breakpoints** with small included angles, george.ee.washington.edu/publications/subjects/./qiangji/pami_corner.ps

[A New Breakpoint Implementation Scheme for Debugging Globally.. - Wu, Hwu \(1998\) \(Correct\)](#)

A New **Breakpoint** Implementation Scheme for Debugging Globally code has become a necessity. Implementing source **breakpoints** is a fundamental aspect of such a debugger. In www.crhc.uiuc.edu/IMPACT/ftp/report/impact-98-06.break.ps

[Breakpoint Medians and Breakpoint Phylogenies: A.. - Gramm, Niedermeier \(2002\) \(Correct\)](#)

Breakpoint Medians and **Breakpoint** Phylogenies: A **Breakpoint** Medians and **Breakpoint** Phylogenies: A Fixed-Parameter Approach Jens www-fs.informatik.uni-tuebingen.de/~niedermr/publications/./eccb02.ps.Z

[Breakpoints and Time in Distributed Computations - Basten \(1994\) \(Correct\) \(3 citations\)](#)

Breakpoints and Time in Distributed Computations Twan investigates how vector time can be used to set **breakpoints** in distributed computations for the purpose of ftp.win.tue.nl/pub/techreports/tbasten/bpts.ps.Z

[Replay and Distributed Breakpoints in an OSF DCE Environment - Yong \(1995\) \(Correct\) \(2 citations\)](#)

Replay And Distributed **Breakpoints** In An Osf Dce Environment By Yuh Ming Yong A such as examining the program states and setting **breakpoints**. In a distributed environment, there is also ccnga.uwaterloo.ca/pub/papers/Ps/thesismas05.ps.Z

[A Novel Breakpoint Implementation Scheme for Debugging Optimized.. - Wu, Whu \(1998\) \(Correct\)](#)

Technical Report IMPACT9801 1 A Novel **Breakpoint** Implementation Scheme for Debugging Optimized such a debugger. In this paper, a novel source **breakpoint** implementation scheme for debugging optimized www.cs.berkeley.edu/~liblit/debug-opt/papers/wu-hwu.ps

[Phylogenies from Gene Order Data: A Detailed Study.. - Moret, Wyman.. \(2001\) \(Correct\)](#)

from Gene Order Data: A Detailed Study of **Breakpoint** Analysis Bernard M.E. Moret, Stacia Wyman, www.cs.utexas.edu/users/phylo/papers/psb.ps

[VASE User's Manual Version 1.0 - David Jablonowski \(Correct\)](#)

:11 3.4 Inserting **Breakpoints** :
:18 5 Using the **Breakpoint** Insertion Tool 19 5.1 Traversing Program
www.csr.d.uiuc.edu/reports/1294.ps.gz

[A New Framework for Debugging Globally Optimized Code - Wu, Mirani, Patil, Olsen, Hwu \(1999\) \(Correct\) \(5 citations\)](#)

recovery of expected variable values at source **breakpoints**. The framework has been prototyped in the reordering. In our debugging framework, for a **breakpoint** at source statement S, the debugger suspends www.cs.wisc.edu/~patil/./docs/pldi99.ps

A Visualization-based Environment for Top-down Debugging of.. - Sharnowski, Cheng (1995) (Correct) (3 citations)
the operation of setting a causal distributed **breakpoint** for a set of processes. Second, a strategy is
do not exist. For example, the ability to set a **breakpoint** distributed across a set of processes is a
<ftp.cps.msu.edu/pub/serg/viz/fpps95.ps>.Z

A Simple and Extensible Graphical Debugger - Hanson, Korn (1997) (Correct) (3 citations)
www.cs.princeton.edu/~drh/pubs/deet.pdf

Detection and Recovery of Endangered Variables Caused by.. - Adl-Tabatabai, Gross (1993) (Correct)
www.cs.cmu.edu/afs/cs.cmu.edu/project/iwarp/archive/fx-papers/pldi93.ps

Hybrid Slicing: An Approach for Refining Static Slices Using.. - Gupta, Soffa (1995) (Correct) (6 citations)
is readily available during debugging, namely **breakpoint** information and the dynamic call graph. The
the potential paths taken by the program. The **breakpoints** and call/return points, used as reference
www.cs.pitt.edu/~gupta/research/SE/fose95.ps

Testing for a Breakpoint in Two-Phase Linear and.. - Gerhard Haybach.. (Correct)
Testing for a **Breakpoint** in Two-Phase Linear and Logistic Regression
at certain unknown points, the so-called **breakpoints**. Thus, testing for the existence of a
ftp.stat.uni-muenchen.de/pub/sfb386/paper77.ps.Z

A Survey of Support For Implementing Debuggers - Paxson (1990) (Correct) (1 citation)
had bugs that took more than a month to find. **Breakpoint** debuggers are now widely available. These
then continue the program. In their usual form **breakpoint** debuggers are adequate for a number of de
the program, trap the resulting **breakpoint**, and remove the successor **breakpoint** [20]Hardware page
<ftp.ee.lbl.gov/papers/debugger-support.ps>.Z

Genomic Sequence Comparison - Pevzner (1995) (Correct)
for $n \geq 200$ are presented in [5]3. **Breakpoint** graph The key idea to sort by reversals is the
to sort by reversals is the definition of the **breakpoint** graph (see Fig. 1)Let $1:n$ be a
\Delta(ae) 1: Proof. sketch)every reversal removes/adds at most two **breakpoints**. One considers all
ftp.inria.fr/INRIA/Projects/algo/web/seminars/sem94-95/pevzner.ps

Evicted Variables and the Interaction of Global Register.. - Ali-Reza Adl-Tabatabai (Correct)
www.cs.cmu.edu/afs/cs.cmu.edu/project/iwarp/archive/fx-papers/popl93.ps

A New Implementation and Detailed Study of Breakpoint.. - Moret, Wyman, Bader.. (2001) (Correct) (3 citations)
A New Implementation and Detailed Study of **Breakpoint** Analysis Bernard M.E. Moret Stacia Wyman
such phylogenetic reconstructions. One method is **breakpoint** analysis, developed by Blanchette and Sankoff
www.smi.stanford.edu/projects/helix/psb01/moret.pdf

Debugging Optimized Code Without Being Misled - Copperman (1993) (Correct) (21 citations)
is a mismatch between where the user expects a **breakpoint** to be located and the **breakpoint's** actual
user expects a **breakpoint** to be located and the **breakpoint's** actual location. A mismatch may occur due to
ftp.cse.ucsc.edu/pub/tr/ucsc-crl-93-21.ps.Z

First 20 documents [Next 20](#)

Try your query at: [Amazon](#) [Barnes & Noble](#) [Google \(RI\)](#) [Google \(Web\)](#) [CSB](#) [DBLP](#)

CiteSeer.IST - Copyright [NEC](#) and [IST](#)

YAHOO!search

"remove breakpoint"

Yahoo! Search

[Advanced](#)
[Preferenc](#)

[Web](#)

[Images](#)









[Directory](#)

[Yellow Pages](#)










[News](#)




[Products](#)

WEB RESULTS 21 - 40 out of about 1,800. Search took 0.15 seconds. ([What's this?](#))

21. [Chapter 12 – Debugging with the Visual J++ Debugger](#) 
 ... right mouse button pop-up menu selection **Insert/Remove Breakpoint**. This option adds a breakpoint at ... menu appears, you can choose to either **Remove Breakpoint** or **Disable Breakpoint** ...
www.tutorialbox.com/tutors/J++/ch12.htm - 35k - [Cached](#)
22. http://sunsite.iisc.ernet.in/virlib/java/vj_unleash/ch12.htm 
 ... right mouse button pop-up menu selection **Insert/Remove Breakpoint**. This option adds a breakpoint at ... menu appears, you can choose to either **Remove Breakpoint** or **Disable Breakpoint** ...
sunsite.iisc.ernet.in/virlib/java/vj_unleash/ch12.htm - 35k - [Cached](#)
23. [GDB Internals](#) 
 ... `insert_breakpoint` and `default_memory_remove_breakpoint` respectively) have been provided so ... `MEMORY_INSERT_BREAKPOINT` and `MEMORY_REMOVE_BREAKPOINT` will likely have instructions that ...
www.delorie.com/gnu/docs/gdb/gdbint_71.html - 50k - [Cached](#)
24. http://www.utdallas.edu/~cantrell/ee6345/4_4BSD-Lite/usr/src/contrib/gdb-4_7.lbl/gdb/tags 
 ... `myaddr, len)$/ adapt_remove_breakpoint remote-adapt.c /^adapt_remove_breakpoint(addr, save)$/ adapt_resume ...`
www.utdallas.edu/~cantrell/ee6345/4_4BSD-Lite/usr/src/contrib/gdb-4_7.lbl/gdb/tags - 234k - [Cached](#)
25. http://examples.oreilly.de/english_examples/palmprog/CDROM/Linux/gcc/gdb-4.16/gdb/breakpoint.c 
 /* Everything about breakpoints, for GDB.
examples.oreilly.de/english_examples/palmprog/CDROM/Linux/gcc/gdb-4.16/gdb/breakpoint.c - 115k - [Cached](#)
26. [Bug 17377 - Add/Remove breakpoint not enabled correctly if selection change from outliner](#) 
 ... **Add/Remove breakpoint** not enabled correctly if selection change from outliner ... the selection between two methods The **add/remove breakpoint** action in the Run menu remains disabled. ...
bugs.eclipse.org/bugs/show_bug.cgi?id=17377 - 16k - [Cached](#)
27. [Using the Debugger](#) 
 ... Choose **Set Breakpoint**, **Remove Breakpoint**, or **Remove All Breakpoints**. ... (Macintosh) to display the context menu, and choose **Breakpoint**, **Remove Breakpoint**, or **Remove All Breakpoints**. ...
www.imagearts.ryerson.ca/abal/lectures/flash/html/19_testing12.html - 6k - [Cached](#)
28. [/usr/share/doc/gdb/changelog.gz](#) 
 ... `arm_rdi_remove_breakpoint`): Rewrite to avoid uninitialized warning ... kill)
 (`arm_rdi_insert_breakpoint`, `arm_rdi_remove_breakpoint`): Likewise. 2002-03-06
 Alexandre Oliva <aoliva@redhat ...

lyre.mit.edu/cgi-bin/dwww?type=file&location=/usr/share/doc/gdb/changelog.gz -
160k - [Cached](#)

29. <http://www.acunia.com/wonka/open-wonka/jpda/jdwp/src/jdwp-eventhandler.c> 
... the breakpoint hashtable. */ w_void remove_breakpoint(jdwp_breakpoint
breakpoint) { w_word ... breakpoint from the hashtable. */ remove_breakpoint
(event->break_point); /* ** Clean up ...
www.acunia.com/wonka/open-wonka/jpda/jdwp/src/jdwp-eventhandler.c - 24k -
[Cached](#)
30. gdb-5.3/gdb/ChangeLog - annotate - 1.1 
Annotation of gdb-5.3/gdb/ChangeLog, Revision 1.1. 1.1 |hilfingr 1: 2002-12-10
GDB Administrator <gdbadmin@sourceware.org> | 2: | 3: * version.in: Bump to
version 5.3. | 4: | 5: 2002-12-10 Andrew Cagney <ac131313@redhat.com> | 6: |
www.libre.act-europe.fr/cvsweb/gdb-5.3/gdb/ChangeLog?annotate=1.1&sortby=log
- 1093k - [Cached](#)
31. [Using the Debugger](#) 
... Choose Set Breakpoint, **Remove Breakpoint**, or Remove All Breakpoints. ...
Macintosh) to display the context menu, and choose Breakpoint, **Remove
Breakpoint**, or Remove All Breakpoints. ...
www.123flashchat.com/flash/19_testing12.html - 19k - [Cached](#)
32. <http://example.oreilly.com/palmprog/CDROM/Linux/gcc/gdb-4.16/gdb/ChangeLog-95> 
all x86 targets and hosts): Add support for pentium-pro machines. * configure:
Rebuild. *
example.oreilly.com/palmprog/CDROM/Linux/gcc/gdb-4.16/gdb/ChangeLog-95 -
201k - [Cached](#)
33. <http://www.magenet.com/ftp/pub/fpc/contrib/libgdb/v5.1.1/gdbsrcpatch> 
Copyright (C) 1997, 1998, 1999, 2001 Free Software Foundation, Inc. + + This file
is part of GDB. + +
www.magenet.com/ftp/pub/fpc/contrib/libgdb/v5.1.1/gdbsrcpatch - 17k - [Cached](#)
34. [Website Templates - Web Design - Web Site Templates Design](#) 
... Choose Set Breakpoint, **Remove Breakpoint**, or Remove All Breakpoints. ...
Macintosh) to display the context menu, and choose Breakpoint, **Remove
Breakpoint**, or Remove All Breakpoints. ...
www.marketingtops.com/tutorial/flash/Flash/html/19_testing12.html - 11k - [Cached](#)
35. <http://www.cis.ufl.edu/research/revcomp/users/sal/jimswing/source/JIMSwingGUI/BreakpointEditDialog.java> 
... removeField.getText(); String command = "REMOVE BREAKPOINT " +
toRemove; crb = commandProcessor.processCommand(command ...
www.cis.ufl.edu/research/revcomp/users/sal/jimswing/source/JIMSwingGUI/BreakpointEditDialog.java
- 5k - [Cached](#)
36. <http://www.online.pl/system-doc/gdb-5.0/gdb/ChangeLog> 
2000-05-17 Andrew Cagney <cagney@sourceware.cygnus.com> * GDB 5.0
released. *
www.online.pl/system-doc/gdb-5.0/gdb/ChangeLog - 107k - [Cached](#)
37. [Line Breakpoints](#) 
... in the Breakpoints view) or **Remove Breakpoint** (in the marker bar). ...

38. [iDLab - Embedded Systems Buffalo Monitor](#) 
... BR -C009. **Remove breakpoint** at \$C009. C003 C005 C007 0000 ...
watchman.idlab.dal.ca/Courses/EmbeddedSystems/Embedded%20Systems%20Design/Bufalo/Bufalobreakpoint. ... - 15k - [Cached](#)
39. [Removing breakpoints](#) 
Removing breakpoints. Breakpoints can be easily removed when you no longer need them. In the editor area, open the file where you want to remove the breakpoint. ... breakpoint, open the marker bar pop-up menu and select **Remove Breakpoint**. The breakpoint is removed from the workbench ...
help.eclipse.org/help21/topic/org.eclipse.jdt.doc.user/tasks/tasks-144a.htm - 3k - [Cached](#)
40. [Class notes on C++](#) 
... Select the "Insert/Remove Breakpoint" option from the menu that appears ...
Select the "**Remove Breakpoint**" option from the menu that appears ...
www.cord.edu/faculty/zhang/cs125/classnotes/visual_cpp_4.htm - 94k - [Cached](#)

Results Page:

[Prev](#) ◀ 1 2 3 4 5 6 7 8 9 10 ▶ [Next](#)

Help us improve your search experience. [Send us feedback](#).

[Web](#)

[Images](#)

[Directory](#)

[Yellow Pages](#)

[News](#)

[Products](#)

Your Search:

[Yahoo! Search](#)

[Advanced Web Search Preferences](#)

Yahoo! Search is hiring! [Learn about job opportunities](#)

Sharing these results with a friend? Use the [Yahoo! Search IMVironment](#)

Copyright © 2004 Yahoo! Inc. All rights reserved. [Privacy Policy](#) - [Terms of Service](#) - [Submit Your Site](#)

